# CPRE

# Certified Professional for Requirements Engineering

Requirements Engineering Glossary

Korean

University of Zurich UZH

Department of Informatics

Requirements Engineering Research Group

RE RG

IREB

## Terms of Use

Individuals and training providers may use this glossary as a basis for seminars, provided that the copyright is acknowledged and included in the seminar materials. Anyone using this glossary in advertising needs the approval of IREB e.V. in writing for this purpose.

Any individual or group of individuals may use this glossary as basis for articles, books or other derived publications provided that this glossary is cited properly.

## About the Author

Martin Glinz is a full professor emeritus at the University of Zurich (UZH). From 1993 until 2017, he was a professor of Informatics at UZH's Department of Informatics. From 2007–2016, he was the department head. His interests include requirements and software engineering — in particular modeling, validation, quality, and evolution.

He received a diploma degree in Mathematics in 1977 and a Dr. rer. nat. in Computer Science in 1983, both from RWTH Aachen University. Before joining the University of Zurich, he worked in industry for ten years, where he was active in software engineering research, development, training, and consulting. He retired in summer 2017, but he is still active in Requirements Engineering research, education, and service.

Martin Glinz has over 35 years of experience in Requirements Engineering, both academic and industrial. He is on editorial boards and program committees of major journals and conferences in software and requirements engineering and served as general chair, program chair, steering committee chair and organizer for the top international conferences in his field. He is a full member of the International Requirements Engineering Board (IREB), where he chairs the IREB Council. He received the ACM SIGSOFT Distinguished Service Award and the IEEE International Requirements Engineering Conference Lifetime Service Award in 2016 and the IEEE International Requirements Engineering Conference Most Influential Paper Award in 2017.

## Preface

In the preface to the first edition of this glossary, published in May 2011, I wrote:

> *When looking for definitions of terms in Requirements Engineering, one can find definitions for almost any term by searching the web. However, such searching requires effort and the quality of the results is unpredictable. Frequently, definitions found in different sources are inconsistent with each other. Existing glossaries in Requirements Engineering textbooks mostly focus on the topics covered in these books. Systematic translations of terminology into major languages other than English are missing completely.*
>
> *This glossary aims at collecting the existing knowledge on Requirements Engineering terminology and defining the core terminology carefully and consistently. In cases where more than one definition is in use or where terms are defined differently when viewed from different perspectives, multiple definitions or perspectives are included. For terms having both a general meaning and a specific meaning in a Requirements Engineering context, both meanings are defined. Important terms are annotated with hints and additional information.*

This glossary has closed the gap identified above. The principle of not just compiling existing definitions but defining the core Requirements Engineering terminology carefully and consistently, has also stood the test of time. Nevertheless, after almost ten years since its initial publication, it was time for a major revision.

A good glossary should be a stable work product: users need to rely on a common terminology — which is not possible when that terminology is constantly changing. On the other hand, it would be foolish to believe that terminology does not evolve over time. In particular, the major revision of the IREB CPRE Foundation Level syllabus required adaptations and extensions of the terminology. Doing a major revision was also an occasion to include important terms from the IREB CPRE Advanced Level syllabi (which did not yet exist when the glossary was initially published). Finally, IREB and ISTQB, the International Software Testing Qualification Board, had agreed in 2019 to harmonize the quality and testing terminology in their respective glossaries.

From the 128 terms defined in the first edition of the glossary, 42 (i.e., about one third) remained unchanged. 67 definitions underwent minor or merely syntactic changes. We re-wrote 17 definitions, deleted two ones, and added 85 new definitions. Major additions concern terminology about agile, modeling, prototyping, and product lines. We also added several basic terms such as activity, method, process, or technique.

Many major changes were due to the harmonization of terminology with ISTQB. However, we also modernized fundamental terms: for example, we simplified the definitions of requirement and Requirements Engineering and made major changes to the notes in the definition of system. The major revision of the glossary was also an occasion to mark explanatory notes clearly in all definitions, separating them from the main definition phrase.

The translations of the terminology into other languages, which were an integral part of the previous versions of this glossary, are now published as separate dictionaries of terminology. I gratefully acknowledge the work performed by all the translators.

Karol Frühauf owes my deepest thanks for carefully reviewing all my definition drafts and for fruitful discussions that led to major improvements of this glossary. I also thank Xavier Franch and Stan Bühne for many helpful comments. Most of all, I thank my wife Angelika. Without her love, patience and understanding, most of my professional work, including this one, would not have been possible.

Martin Glinz

Zurich, October 2020

## Acknowledgements

## Translation

## CPRE Online Glossary

The CPRE Glossary is available online in all supported languages:
https://www.ireb.org/en/cpre/glossary/

## Version History

| Version | Date | Change |
|---------|------|--------|
| 2.1.0 | March 2025 | Initial Document |

# Contents

# 1 Definitions of Terms

Terms formatted in **bold** are key terms that have to be known on the IREB CPRE Foundation Level.

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 개체 | Entity | <ul><li>In general:<ul><li>Anything which is perceivable or conceivable (→ item).</li></ul></li><li>In entity–relationship-modeling:<ul><li>an individual ↑item which has an identity and does not depend on another item (→ object).</li></ul></li></ul> |
| 개체-관계 다이어그램 | Entity-relationship diagram | A diagrammatic representation of an ↑entity-relationship model. <br>**Abbreviation:** ERD |
| 개체–관계 모델 | Entity-relationship model | A ↑model of data that are relevant for a ↑system or of the data of an ↑application domain, consisting of a set of entity types that are each characterized by ↑attributes and linked by relationships. <br>**Abbreviation:** ER Model |
| **객체** | **Object** | <ul><li>In general:<ul><li>Anything which is perceivable or conceivable (→ item).</li></ul></li><li>In software engineering:<ul><li>an individual ↑item which has an identity, is characterized by the values of its ↑attributes and does not depend on another item (→ entity).</li></ul></li></ul> |
| 객체 다이어그램 | Object diagram | A diagrammatic representation of an ↑object model. |
| **객체 모델** | **Object model** | A ↑model describing a set of ↑objects and relationships between them. |
| (요구사항의) **검증 가능성** | **Verifiability** (of requirements) | The degree to which the fulfillment of a ↑requirement by an implemented ↑system can be verified. <br>**Note:** <br>Such ↑verification can be performed, for example, by defining ↑acceptance test cases, measurements or ↑inspection procedures. |

| 용어 (한국어) | Term (English) | Definition |
| --- | --- | --- |
| 결정 테이블 | Decision table | A tabular representation of a complex decision, specifying which actions to perform for the possible combinations of condition values. |
| 결함 | **Defect** | An imperfection or deficiency in a ↑work product that impairs its intended use.<br>**Synonyms:** bug, fault |
| 고객 | **Customer** | A person or organization who receives a ↑system, a ↑product or a ↑service.<br>Also see ↑stakeholder. |
| 고객 요구사항 명세 | **Customer requirements specification** | A coarse description of the required capabilities of a ↑system from the ↑customer's perspective.<br>**Note:**<br>A customer requirements specification is usually supplied by the ↑customer. |
| 공급자 | Supplier | A person or organization who delivers a ↑product or ↑service to a ↑customer. |
| 공통점 | Commonality | The parts of a ↑product line that are shared by all its members. |
| 구문 | **Syntax** | The rules for constructing structured signs in a ↑language. |
| 구문 템플릿 | Phrase template | A template for the syntactic structure of a phrase that expresses an individual ↑requirement or a ↑user story in ↑natural language. (→ requirements template) |
| (기술적 맥락에서의) 구성 | Composition (in a technical context) | ▪ An ↑item that is composed of a set of items; forming a whole–part relationship.<br>▪ The act of composing a whole from a set of parts. |
| 구조화된 분석 | Structured Analysis | An approach for specifying the ↑functionality of a system based on a hierarchy of ↑data flow diagrams. Data flows as well as persistent data are defined in a data dictionary. A ↑context diagram models the sources of incoming and the destinations of outgoing ↑data flows. |
| 규정 준수 | **Compliance** | The adherence of a ↑work product to ↑standards, conventions, regulations, laws, or similar prescriptions. |
| 기능 요구사항 | **Functional requirement** | A ↑requirement concerning a result or ↑behavior that shall be provided by a function of a ↑system. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| **기능성** | **Functionality** | The capabilities of a ↑system as stated by its ↑functional requirements. |
| 기법 | Technique | A documented set of coherent actions for accomplishing a ↑task or achieving an objective. |
| **기준** | **Baseline** | A stable, change-controlled ↑configuration of ↑work products.<br>**Note:**<br>Baselines serve for ↑release planning and release definition as well as for project management purposes such as effort estimation. |
| **네이티브 프로토타입** | **Native prototype** | A high-fidelity ↑prototype that implements critical parts of a ↑system to an extent that ↑stakeholders can use the prototype to see whether the prototyped part of the system will work and behave as expected. |
| 다중도 | Multiplicity | → Cardinality |
| 데이터 흐름 | Data flow | A sequence of data items flowing from a producer to a consumer. |
| 데이터 흐름 다이어그램 | Data flow diagram | A diagrammatic representation of a ↑data flow model.<br>**Abbreviation:** DFD |
| 데이터 흐름 모델 | Data flow model | A model that describes the ↑functionality of a ↑system by ↑activities, data stores and ↑data flows.<br>**Note:**<br>Incoming data flows trigger activities which then consume the received data, transform them, read/write persistent data held in data stores and then produce new data flows which may be intermediate results that trigger other activities or final results that leave the system. |
| (소프트웨어 공학에서의)<br>**도구** | **Tool** (in software engineering) | A (software) ↑system that helps develop, operate and maintain systems.<br>**Note:**<br>In RE, tools support ↑requirements management as well as modeling, documenting, and validating ↑requirements. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 도메인 | **Domain** | A range of relevant things (for some given matter); for example, an ⬆application domain. |
| 도메인 모델 | **Domain model** | A ⬆model describing phenomena in an ⬆application domain.<br>**Note:**<br>▪ In RE, domain models are created with the intention to understand the ⬆application domain in which a planned ⬆system will be situated.<br>▪ Static domain models specify (business) objects and their relationships in a ⬆domain of interest.<br>▪ Domain story models specify visual stories about how actors interact with devices, artifacts, and other items in a ⬆domain. |
| 도메인 요구사항 | **Domain requirement** | A ⬆domain property in the ⬆context of a ⬆system that is required to hold. |
| (요구사항의) 도출 | **Elicitation** (of requirements) | → Requirements elicitation |
| 동음이의어 | Homonym | A term looking identical to another term but having a different meaning.<br>**Note:**<br>For example, bill as a bank note and bill as a list (of materials) are homonyms. |
| 동의어 | **Synonym** | A word having the same meaning as another word. |
| 동작 | **Behavior** | The way in which a ⬆system reacts to stimuli, changes its state and produces observable results.<br>**Note:**<br>Stimuli may be events or changes of conditions. Their origin may be external or system-internal. |
| 동작 모델 | **Behavior model** | A ⬆model describing the ⬆behavior of a ⬆system, e.g., by a ⬆state machine. |
| 리뷰 | **Review** | An evaluation of a ⬆work product by an individual or a group in order to find problems or suggest improvements.<br>**Note:**<br>Evaluation may be performed with respect to both contents and conformance. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 리스크 | **Risk** | A possible event that threatens the success of an endeavor.<br>**Note:**<br>A risk is typically assessed in terms of its probability and potential damage. |
| 리팩토링 | Refactoring | Multiple occurrence of the same information or resource. |
| 릴리스 | **Release** | A ⬆configuration that has been released for installation and use by ⬆customers. |
| (요구사항의) **명확성** | **Unambiguity** (of requirements) | The degree to which a ⬆requirement is expressed such that it cannot be understood differently by different people. |
| 명세 | **Specification** | ▪ As a work product:<br> ▪ A systematically represented description of the properties of an ⬆item (a ⬆system, a device, etc.) that satisfies given criteria.<br>▪ As a process:<br> ▪ the process of specifying (⬆eliciting, documenting and ⬆validating) the properties of an ⬆item.<br><br>**Note:**<br>A specification may be about required properties (⬆requirements specification) or implemented properties (e.g., a technical product specification). |
| 명세 언어 | Specification language | An artificial ⬆language that has been created for expressing ⬆specifications. |
| 모델 | **Model** | An abstract representation of an existing part of reality or a part of reality to be created.<br>**Note:**<br>▪ The notion of reality includes any conceivable set of elements, phenomena, or concepts, including other models.<br>▪ Models are always built for *specific purposes* in a *specific context*.<br>▪ With respect to a model, the modeled part of reality is called the *original*.<br>▪ In RE, ⬆requirements can be specified with models. |
| 모델 언어 | **Modeling language** | A ⬆language for expressing ⬆models of a certain kind. May be textual, graphic, symbolic or some combination thereof. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| (디지털 시스템의) **목업** | **Mock-up** (of a digital system) | A medium-fidelity ↑prototype that demonstrates characteristics of a user interface without implementing any real ↑functionality.<br>**Note:**<br>In RE, a mock-up primarily serves for specifying and validating user interfaces. |
| 모호성 | Ambiguity | The contrary of →unambiguity |
| **목표** | **Goal** | A desired state of affairs (that a ↑stakeholder wants to achieve).<br>**Note:**<br>Goals describe intentions of stakeholders. They may conflict with one another. |
| **목표 모델** | **Goal model** | A ↑model representing a set ↑goals, sub-goals and the relationships between them.<br>**Note:**<br>Goal models may also include tasks and resources needed to achieve a goal, actors who want to achieve a goal, and obstacles that impede the achievement of a goal. |
| 문서 템플릿 | Document template | A template providing a predefined skeleton structure for a document. (→ requirements template)<br>**Note:**<br>In RE, document templates can be used to structure ↑requirements documents. |
| 문제 | Problem | A difficulty, open question or undesirable condition that needs investigation, consideration, or solution. |
| **반복** | **Iteration** | ▪ In general:<br>  ▪ The repetition of something, for example, a procedure, a process or a piece of program code.<br>▪ In agile development:<br>  ▪ A ↑timeboxed unit of work in which a development team implements an ↑increment to the ↑system under development.<br>**Note:**<br>In agile development, iteration and ↑sprint are frequently used as synonyms. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 반-정형적 | Semi-formal | Something which is formal to some extent, but not completely.<br>**Note:**<br>A ↑work product is called semi-formal if it contains formal parts, but isn't formalized totally. Typically, a semi-formal work product has a defined ↑syntax, while the ↑semantics is partially defined only. |
| 방법 | Method | The systematic application of a ↑technique (or a set of techniques) to achieve an objective or create a ↑work product. |
| 방법론 | Methodology | • The systematic study of ↑methods in a particular field, in particular, how to select, apply or evaluate methods systematically in a given situation.<br>• A set of ↑methods being applied in some combination. |
| 백로그 | Backlog | → Product backlog, → sprint backlog |
| **유효성 검증** | **Validation** | The ↑process of confirming that an ↑item (a ↑system, a ↑work product or a part thereof) matches its ↑stakeholders' needs.<br>**Note:**<br>In RE, validation is the process of confirming that the documented ↑requirements match their ↑stakeholders' needs; in other words: whether the right requirements have been specified. |
| 버그 | Bug | → Defect |
| **버전** | **Version** | An occurrence of an ↑item which exists in multiple, time-ordered occurrences where each occurrence has been created by modifying one of its previous occurrences. |
| 번다운 차트 | Burndown chart | A diagram plotting the work items that remain to accomplish on a time scale. |
| (시스템 개발의) **범위** | **Scope** (of a system development) | The range of things that can be shaped and designed when developing a ↑system. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 베리피케이션 | Verification | The process of confirming that an ↑item (a system, a work product, or a part thereof) fulfills its ↑specification.<br>**Note:**<br>Requirements verification is the process of confirming that the ↑requirements have been documented properly and satisfy the ↑quality criteria for requirements; in other words, whether the requirements have been specified right. |
| 변경 가능성 | Changeability | → Modifiability |
| **변경 관리** | **Change management** | A controlled way to effect or deny a requested change of a ↑work product. |
| **변경 요청** | **Change request** | In RE: A well-argued request for changing one or more ↑baselined ↑requirements. |
| **변경 제어 위원회** | **Change control board** | A committee of ↑customer and ↑supplier representatives that decides on ↑change requests.<br>**Abbreviation:** CCB<br>**Note:**<br>The Change control board should not be confused with a *change advisory board*, which is a committee that evaluates change requests for a ↑system in operation and typically has no decision power. |
| 변동성 | Variability | ▪ The degree to which a ↑system can be changed or customized.<br>▪ In product lines: The ↑features that can differ among the members of the ↑product line. |
| 변형 | Variant | One of the possible forms that an ↑item (e.g., a ↑requirement) may have. |
| 변형점 | Variation point | A point in a ↑product line where an element of the product line (typically a variable or a ↑feature) can be chosen from a set of ↑variants. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 보안 | **Security** | The degree to which a ⬆system protects its data and resources against unauthorized access or use and secures unobstructed access and use for its legitimate ⬆users.<br>**Note:**<br>Security requirements may be stated as ⬆quality requirements or in terms of ⬆functional requirements. |
| 분기 | Branch | A line of ⬆configurations or ⬆work product ⬆versions that forks away from the main line (or from another branch) at some point in time.<br>**Note:**<br>A branch is created by making a copy of some configuration or work product version and making this copy the root of the branch. A branch may be merged with the main line or with another branch at some later point in time. |
| 뷰 | **View** | An excerpt from a ⬆work product, containing only those parts one is currently interested in.<br>**Note:**<br>A view can abstract or aggregate parts of the work product. |
| 뷰포인트, 시각 | Viewpoint | A certain perspective on the ⬆requirements of a ⬆system.<br>**Note:**<br>Typical viewpoints are perspectives that a ⬆stakeholder or stakeholder group has (for example, an end user's perspective or an operator's perspective). However, there can also be topical viewpoints such as a security viewpoint. |
| 비기능 요구사항 | **Non–functional requirement** | A ⬆quality requirement or a ⬆constraint.<br>**Note:**<br>⬆Performance requirements may be regarded as another category of non–functional requirements. In this glossary, performance requirements are considered to be a sub–category of ⬆quality requirements. |
| (시스템이나 제품의) 비전 | **Vision** (for a system or product) | A conceptual imagination of a future ⬆system or ⬆product, describing its key characteristics and how it will create value for its ⬆users. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 비즈니스 요구사항 | **Business requirement** | A ⬆requirement stating a business ⬆goal, objective or need of an organization.<br>**Note:**<br>Business requirements typically state those business goals, objectives and needs that shall be achieved by employing a ⬆system or a collection of systems. |
| 사용성 | **Usability** | The degree to which a ⬆system can be used by specified ⬆users to achieve specified ⬆goals in a specified context of use.<br>**Note:**<br>Usability particularly includes the capability of a ⬆system to be understood, learned, used, and liked by its intended ⬆users. |
| 사용자 | **User** | A person who uses the ⬆functionality provided by a ⬆system.<br>**Note:**<br>Users (also called end users) always are ⬆stakeholders of a ⬆system. |
| 사용자 스토리 | **User story** | A description of a need from a ⬆user's perspective together with the expected benefit when this need is satisfied.<br>**Note:**<br>▪ User stories are typically written in ⬆natural language using a ⬆phrase template and are accompanied by ⬆acceptance criteria.<br>▪ In ⬆agile development, user stories are the main means for communicating needs between a ⬆product owner and the development team. |
| 사용자 요구사항 | **User requirement** | A ⬆requirement expressing a ⬆user need.<br>**Note:**<br>User requirements are typically about what a system should do for certain users and how they can interact with the system. User requirements are a subset of ⬆stakeholder requirements. |
| 산출물 | Artifact | Synonym for ⬆work product. |
| (요구사항의) 상세화 | **Elaboration** (of requirements) | An umbrella term for requirements ⬆elicitation, ⬆negotiation and ⬆validation. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 상태 머신 | **State machine** | A ⬆model describing the behavior of a ⬆system by a finite set of *states* and state *transitions*. State transitions are triggered by *events* and can in turn trigger *actions* and new events. |
| 상태 머신 다이어그램 | **State machine diagram** | A diagrammatic representation of a ⬆state machine. |
| 상태 전이 다이어그램 | State-transition diagram | A ⬆state machine having states that are hierarchically and/or orthogonally decomposed. |
| 상태 차트 | **Statechart** | → State machine diagram |
| 서비스 | **Service** | The provision of some ⬆functionality to a human or a ⬆system by a provider (a system, organization, group or individual) that delivers value to the receiver.<br>**Note:**<br>In systems engineering, software engineering and Requirements Engineering, services are typically provided by a ⬆system for a ⬆user or another system. |
| 서식 | Form template | A template providing a form with predefined fields to be filled-in. (→ requirements template)<br>**Note:**<br>In RE, form templates can be used to specify ⬆use cases or ⬆quality requirements. |
| 설계 | **Design** | <ul><li>A plan or drawing produced to show how something will look, function or be structured before it is made.</li><li>The activity of creating a design.</li><li>A decorative pattern [This meaning does not apply in the software engineering ⬆domain].</li></ul>**Note:**<ul><li>In software product development, we distinguish between *creative design* which shapes the look and feel of the product, i.e., its perceivable form, function and quality, and *technical design* (also called software design) which determines the inner structure of the product, in particular the software architecture.</li><li>The creative design of products is also called *product design*.</li><li>The creative design of digital solutions is called *digital design*.</li></ul> |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 성능 요구사항 | **Performance requirement** | A ↑requirement describing a performance characteristic (timing, speed, volume, capacity, throughput, …).<br>**Note:**<br>In this glossary, performance requirements are regarded as a sub–category of ↑quality requirements. However, they can also be considered as a ↑kind of requirements of its own. |
| (요구사항의) 출처 | Source (of a requirement) | → Requirements source |
| 소프트웨어 요구사항 명세 | **Software requirements specification** | A ↑requirements specification pertaining to a software ↑system.<br>**Abbreviation:** SRS |
| 속성 | **Attribute** | A characteristic property of an ↑entity or an ↑object. |
| 수정 가능성 | **Modifiability** | The degree to which a ↑work product or ↑system can be modified without degrading its ↑quality. |
| 시퀀스 다이어그램 | **Sequence diagram** | A diagram type in ↑UML which models the interactions between a selected set of ↑objects and/or ↑actors in the sequential order in which those interactions occur. |
| 스크럼 | Scrum | A popular ↑process framework for ↑agile development of a ↑system. |
| (요구공학에서의) 스토리 | Story (in an RE context) | → User story |
| 스토리 맵 | Story map | A two–dimensional arrangement of ↑user stories.<br>**Note:**<br>A story map helps understand the ↑functionality of a ↑system, identify gaps and plan releases. |
| 스토리보드 | Storyboard | A series of sketches or pictures that visualize the execution of a ↑scenario. |
| 스파이크 | Spike | In agile development: A task aimed at gaining insight or gathering information, rather than at producing a ↑product ↑increment. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 스프린트 | **Sprint** | An ⬆iteration in ⬆agile development, particularly when using ⬆Scrum. |
| 스프린트 백로그 | Sprint backlog | A set of ⬆product backlog items that have been selected to be implemented in the current ⬆sprint. |
| 시나리오 | Scenario | <ul><li>In general:<ul><li>A description of a potential sequence of events that lead to a desired (or unwanted) result.</li></ul></li><li>In RE:<ul><li>An ordered sequence of interactions between partners, in particular between a ⬆system and external ⬆actors. May be a concrete sequence (instance scenario) or a set of potential sequences (type scenario, ⬆use case).</li></ul></li></ul> |
| 시스템 | **System** | <ul><li>In general:<ul><li>A principle for ordering and structuring.</li></ul></li><li>In engineering:<ul><li>A coherent, delimitable set of elements that – by coordinated action – achieve some purpose.</li></ul></li></ul>**Note:**<ul><li>A system may comprise other systems or ⬆components as sub–systems.</li><li>The purposes achieved by a system may be delivered by<ul><li>deploying the system at the place(s) where it is used,</li><li>selling/providing the system as a ⬆product to its ⬆users,</li><li>having providers who offer the system's capabilities as ⬆services to users.</li></ul></li><li>Systems containing both software and physical ⬆components are called *cyber–physical systems*.</li><li>Systems spanning software, hardware, people and organizational aspects are called *socio–technical systems*.</li></ul>**Important:** In all definitions referring to system in this glossary, system is an umbrella term which includes<ul><li>⬆*Products* provided to ⬆customers,</li><li>⬆*Services* made available to ⬆customers,</li><li>Other work products such as *devices*, *procedures* or *tools* that help people or organizations achieve some goal,</li><li>System ⬆*components* or ⬆*compositions* of systems.</li></ul> |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 시스템 경계 | **System boundary** | The boundary between a ⬆system and its surrounding ⬆context.<br>**Note:**<br>▪ The system boundary delimits the system as it shall be after its implementation and deployment.<br>▪ At the system boundary, the external interfaces between the ⬆system and its ⬆context have to be defined.<br>▪ The system boundary frequently coincides with the ⬆scope of a ⬆system (which denotes the range of things that can be shaped and designed). However, this is not always the case: there may be components within the system boundary that have to be re-used as they are (i.e., cannot be shaped nor designed), while in the system context there may be things that can be re-designed when the system is developed (which means that they are in scope). |
| 시스템 요구사항 | **System requirement** | A ⬆requirement pertaining to a ⬆system. |
| 시스템 요구사항 명세 | **System requirements specification** | A ⬆requirements specification pertaining to a ⬆system.<br>**Note:**<br>A system requirements specification is frequently considered to be a synonym for ⬆requirements specification.<br>**Abbreviation:** SyRS |
| 시스템 정황 | System context | The part of a ⬆system's environment that is relevant for the definition as well as the understanding of the ⬆requirements of a ⬆system to be developed. |
| 신뢰성 | **Reliability** | The degree to which a ⬆system performs specified functions under specified conditions for a specified period of time.<br>**Note:**<br>Reliability may be stated as a ⬆quality requirement. |
| 실천법 | Practice | A proven way of how to carry out certain types of ⬆tasks or ⬆activities. |
| 오류 | **Fault** | → Defect |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 오류 내구성 | **Fault tolerance** | The capability of a ⬆system to operate as intended despite the presence of (hardware or software) ⬆faults.<br>**Note:**<br>Fault tolerance may be stated as a ⬆quality requirement. |
| 안전 | **Safety** | The capability of a ⬆system to achieve an acceptable level of probability that the system, under defined conditions, will not reach a state in which human life, health, property, or the environment is endangered.<br>**Note:**<br>Safety ⬆requirements may be stated as ⬆quality requirements or in terms of ⬆functional requirements. |
| 애자일 | **Agile** | ▪ In general:<br>　▪ Able to move quickly and easily.<br>　▪ Quick, smart, and clever.<br>▪ In software development:<br>　▪ A development approach which builds a product ⬆incrementally by dividing work into ⬆iterations of fixed duration (⬆timeboxes).<br><br>**Note:**<br>Agile development is characterized by focusing on delivering a working product in each iteration, collaboration with ⬆stakeholders with frequent feedback and adaptation of plans after each iteration based on feedback and changed ⬆requirements. |
| 애플리케이션 도메인 | **Application domain** | Those parts of the real world that are relevant for determining the ⬆context of a ⬆system. |
| 언어 | Language | A structured set of signs for expressing and communicating information.<br>**Note:**<br>Signs are any elements that are used for communication: spoken or written words or expressions, symbols, gestures, sounds, etc. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 에러 | **Error** | ▪ A human action that produces an incorrect result.<br>▪ A discrepancy between an observed ⬆behavior or result and the specified behavior or result.<br>**Note:**<br>In practice, both meanings are used. Where needed, the meaning of error can be disambiguated by using human error and observed error or observed fault, respectively. |
| 에픽 | Epic | In agile development: An abstract description of a ⬆stakeholder need which is larger than what can be implemented in a single ⬆iteration. |
| 액터 | **Actor** | A person in some ⬆role, a ⬆system or a technical device in the context of a subject under consideration that interacts with that subject.<br>**Note:**<br>In RE, the subject under consideration typically is a ⬆system. In testing, it may be a test ⬆object. |
| 역할 | **Role** | ▪ In general:<br>　▪ A part played by a person in a given context.<br>▪ In ⬆UML ⬆class models:<br>　▪ The parts played by the linked ⬆objects in an ⬆association. |
| 예시 명세 | Specification by example | A ⬆technique that specifies test cases and ⬆requirements for a ⬆system by providing examples of how the system should behave. |
| 와이어어프레임 | **Wireframe** | A low-fidelity ⬆prototype built with simple materials that primarily serves for discussing and validating requirements, design ideas or user interface concepts.<br>**Note:**<br>When prototyping digital systems, wireframes are typically built with paper. Such prototypes are also called *paper prototypes*. |
| (요구사항의) **완성도** | **Completeness** (of requirements) | ▪ For a single ⬆requirement:<br>　▪ The degree to which the specification of a requirement is self-contained.<br>▪ For a ⬆work product covering multiple requirements:<br>　▪ The degree to which the work product contains all known requirements that are relevant in the scope of this work product. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 요구공학 | **Requirements Engineering** | The systematic and disciplined approach to the ⬆specification and management of ⬆requirements with the goal of understanding the ⬆stakeholders' desires and needs and minimizing the risk of delivering a ⬆system that does not meet these desires and needs. **Abbreviation:** RE |
| 요구사항 | **Requirement** | ▪ A need perceived by a ⬆stakeholder.<br>▪ A capability or property that a ⬆system shall have.<br>▪ A documented representation of a need, capability or property. |
| 요구사항 관리 | **Requirements management** | The process of managing existing ⬆requirements and requirements–related ⬆work products, including the storing, changing and tracing of requirements (⬆traceability). |
| 요구사항 명세 | **Requirements specification** | A systematically represented collection of ⬆requirements, typically for a ⬆system, that satisfies given criteria.<br>**Note:**<br>▪ In some situations we distinguish between a ⬆*customer* requirements specification (typically written by the ⬆customer) and a ⬆*system* requirements specification or ⬆*software* requirements specification (written by the supplier).<br>▪ Requirements specification may also denote the ⬆*process* of specifying (⬆eliciting, documenting and ⬆validating) requirements. |
| 요구사항 모델 | Requirements model | A ⬆model that has been created with the purpose of specifying ⬆requirements. |
| 요구사항 문서 | **Requirements document** | A document consisting of a ⬆requirements specification.<br>**Note:**<br>Requirements document is frequently used as a synonym for requirements specification. |
| 요구사항 베이스라인 | Requirements baseline | A ⬆baseline for a set of ⬆requirements. |
| 요구사항 분기 | Requirements branching | → Branch |
| 요구사항 분석 | Requirements analysis | ▪ Analysis of elicited ⬆requirements in order to understand and document them.<br>▪ Synonym for ⬆Requirements Engineering. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 요구사항 출처 | **Requirements source** | The source from which a ↑requirement has been derived.<br>**Note:**<br>Typical sources are ↑stakeholders, documents, existing ↑systems and observations. |
| 요구사항 식별 | Requirements discovery | → Requirements elicitation |
| 요구사항 엔지니어 | **Requirements Engineer** | A person who – in collaboration with ↑stakeholders – elicits, documents, validates, and manages ↑requirements.<br>**Note:**<br>In most cases, requirements engineer is a ↑role and not a job title. |
| 요구사항 도출 | **Requirements elicitation** | The process of seeking, capturing and consolidating ↑requirements from available ↑sources, potentially including the re–construction or creation of requirements. |
| 요구사항 충돌 | **Requirements conflict** | ▪ A situation where two or more ↑requirements cannot be satisfied together.<br>▪ A situation where two or more ↑stakeholders disagree about certain ↑requirements.<br>**Note:**<br>Requirements conflicts have to be solved by ↑requirements negotiation. |
| 요구사항 템플릿 | **Requirements template** | A template for specifying ↑requirements.<br>**Note:**<br>In RE, several forms of templates are used. ↑*Phrase templates* are used for specifying individual ↑requirements or ↑user stories. ↑*Form templates* can be used to specify ↑use cases or ↑quality requirements. ↑*Document templates* provide a predefined structure for ↑requirements documents. |
| 요구사항 협상 | **Requirements negotiation** | A ↑process where ↑stakeholders are working toward reaching an agreement to resolve ↑requirements conflicts. |
| 요구사항 구성 | Requirements configuration | → Configuration |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 요구사항의 종류 | **Kind of requirement** | A classification of requirements according to their kind into ⬆*system* requirements (consisting of ⬆*functional* requirements, ⬆*quality* requirements and ⬆*constraints*), *project* requirements, and *process* requirements.<br>**Note:**<br>▪ RE is primarily concerned with system requirements.<br>▪ Quality requirements and constraints are also called ⬆*non-functional* requirements. |
| 용어집 | **Glossary** | A collection of definitions of terms that are relevant in some ⬆domain.<br>**Note:**<br>Frequently, a glossary also contains cross-references, ⬆synonyms, ⬆homonyms, acronyms, and abbreviations. |
| 우선순위 | **Priority** | The level of importance assigned to an ⬆item, e.g., a ⬆requirement or a ⬆defect, according to certain criteria. |
| 우선순위 지정 | Prioritization | The process of assigning priorities to a set of ⬆items. |
| 운영 위원회 | Steering committee | A committee that supervises a project. |
| 워크스루 | **Walkthrough** | A ⬆review in which the author of a ⬆work product leads the reviewers systematically through the work product and the reviewers ask questions and make comments about possible issues. |
| 유스케이스 | **Use case** | A set of possible interactions between external ⬆actors and a ⬆system that provide a benefit for the actor(s) involved.<br>**Note:**<br>Use cases specify a system from a user's (or other external actor's) perspective: every use case describes some ⬆functionality that the system must provide for the actors involved in the use case. |
| 유스케이스 다이어그램 | **Use case diagram** | A diagram type in ⬆UML that models the ⬆actors and the ⬆use cases of a ⬆system.<br>**Note:**<br>The boundary between the actors and the use cases constitutes the ⬆system boundary. |

| 용어 (한국어) | Term (English) | Definition |
| --- | --- | --- |
| 유스케이스 모델 | Use case model | A ↑model consisting of a set of ↑use cases, typically together with a ↑use case diagram. |
| 유지보수성 | Maintainability | The ease with which a ↑system can be modified by the intended maintainers.<br>**Note:**<br>Maintainability may be stated as a ↑quality requirement. |
| 의미 | **Semantics** | The meaning of a sign or a set of signs in a ↑language. |
| 이동성 | Portability | The ease with which a ↑system can be transferred to another platform while preserving its characteristics. |
| 이해 가능성 | **Understandability** | The degree to which an ↑item is comprehensible to its intended users.<br>**Note:**<br>Typical items are: a ↑system, a ↑work product, or a part thereof. |
| 이해관계자 | **Stakeholder** | A person or organization who influences a ↑system's ↑requirements or who is impacted by that system.<br>**Note:**<br>Influence can also be indirect. For example, some stakeholders may have to follow instructions issued by their managers or organizations. |
| 이해관계자 요구사항 | **Stakeholder requirement** | A ↑requirement expressing a ↑stakeholder desire or need.<br>**Note:**<br>Stakeholder requirements are typically written by stakeholders and express their desires and needs from their perspective. |
| 인수 | Acceptance | The process of assessing whether a ↑system satisfies all its ↑requirements. |
| 인수 조건 | **Acceptance criteria** | In agile: Criteria that the implementation of a ↑user story must satisfy in order to be accepted by the ↑stakeholders.<br>**Note:**<br>Acceptance criteria may also be written for ↑backlog items other than user stories. |

| 용어 (한국어) | Term (English) | Definition |
| --- | --- | --- |
| 인수 테스트 | Acceptance test | A test that assesses whether a ⬆system satisfies its ⬆requirements.<br>**Note:**<br>Typically used by ⬆customers to determine whether or not to accept a system. |
| 인스펙션 | **Inspection** | A formal ⬆review of a ⬆work product by a group of experts according to given criteria, following a defined procedure. |
| (요구사항의) **일관성** | **Consistency** (of requirements) | The degree to which a set of ⬆requirements is free of contradicting statements. |
| 자연어 | **Natural language** | A ⬆language that people use for speaking and writing in everyday life.<br>**Note:**<br>This is in contrast to *artificial languages* that people have deliberately created for specific purposes such as programming or specifying. |
| 작업 산출물 | **Work product** | A recorded, intermediate or final result generated in a work ⬆process.<br>**Synonym:** ⬆Artifact |
| (요구사항의) **적절성** | **Adequacy** (of a requirement) | The degree to which a ⬆requirement expresses the ⬆stakeholders' true and agreed desires and needs (i.e., those they had actually in mind when stating the requirement). |
| 적합성 | **Conformity** | The degree to which a ⬆work product conforms to regulations given in some ⬆standard. |
| 정확성 | Correctness | The degree to which the information contained in a ⬆work product is provably true.<br>**Note:**<br>In RE, correctness is sometimes used as a synonym for ⬆adequacy, particularly when validating a ⬆requirement rigorously against formally stated properties in the ⬆context of a ⬆system. |
| (요구사항에서의) 제약사항 | **Constraint** (in RE) | A ⬆requirement that limits the solution space beyond what is necessary for meeting the given ⬆functional requirements and ⬆quality requirements. |
| 제어 흐름 | Control flow | The order in which a set of actions is executed. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| (소프트웨어에서의) 제품 | **Product** (in the context of software) | A software-based ⬆system or a ⬆service provided by a system which is developed and marketed by a ⬆supplier and used by ⬆customers. |
| 제품 백로그 | **Product backlog** | An ordered, typically prioritized collection of work items that a development team has to work on when developing or evolving a ⬆system.<br>**Note:**<br>Items include ⬆requirements, ⬆defects to be fixed, or ⬆refactorings to be done. |
| 제품 소유자 | Product owner | A person responsible for a ⬆product in terms of ⬆functionality, value and ⬆risk.<br>**Note:**<br>The product owner maintains and prioritizes the ⬆product backlog, makes sure that the ⬆stakeholders' ⬆requirements as well as market needs are elicited and adequately documented in the ⬆product backlog and represents the stakeholders when communicating with the development team. |
| 중복 | **Redundancy** | The improvement of the internal ⬆quality of source code, particularly the structure of the code, without changing its observable behavior. |
| (소프트웨어 개발에서의) 증분 | **Increment** (in software development) | An addition to a ⬆system under development that extends, enhances or refactors (⬆refactoring) the existing parts of the system.<br>**Note:**<br>In ⬆agile development, every ⬆iteration produces an increment. |
| 진화적 프로토타입 | **Evolutionary prototype** | A pilot system forming the core of a ⬆system to be developed. |
| 최종 사용자 | End user | → User |
| 추적성 | **Traceability** | ▪ In general:<br>  ▪ The ability to establish explicit relationships between related ⬆work products or ⬆items within work products.<br>▪ In RE:<br>  ▪ The ability to trace a ⬆requirement<br>    ▪ back to its origins,<br>    ▪ forward to its implementation in design and code and its associated tests,<br>    ▪ to requirements it depends on (and vice-versa). |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| (요구사항에 대한) 충돌 | Conflict (about requirements) | → Requirements conflict |
| 카디널리티 | Cardinality | - In modeling:<br>  - The minimum and maximum number of ↑objects in a relationship.<br>- In mathematics:<br>  - The number of elements in a set.<br><br>**Note:**<br>In ↑UML, the term multiplicity is used for cardinality. |
| 컨텍스트(정황) | **Context** | - In general:<br>  - The network of thoughts and meanings needed for understanding phenomena or utterances.<br>- Especially in RE:<br>  - The part of a ↑system's environment being relevant for understanding the system and its ↑requirements.<br><br>**Note:**<br>1) Context in the second meaning is also called the ↑system context. |
| 컨텍스트(정황) 경계 | **Context boundary** | The boundary between the ↑context of a ↑system and those parts of the ↑application domain that are irrelevant for the ↑system and its ↑requirements.<br>**Note:**<br>The context boundary separates the relevant part of the environment of a system to be developed from the irrelevant part, i.e., the part that does not influence the system to be developed and, thus, does not have to be considered during Requirements Engineering. |
| 컨텍스트 다이어그램 | **Context diagram** | - A diagrammatic representation of a ↑context model.<br>- In ↑Structured Analysis, the context diagram is the root of the ↑dataflow diagram hierarchy. |
| 컨텍스트 모델 | **Context model** | A ↑model describing a ↑system in its ↑context. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 컴포넌트 | **Component** | ▪ In general:<br>　　▪ A delimitable part of a ⬆system.<br>▪ In software architecture:<br>　　▪ An encapsulated set of coherent ⬆objects or ⬆classes that jointly achieve some purpose.<br>▪ In testing:<br>　　▪ A part of a ⬆system that can be tested in isolation.<br><br>**Note:**<br>When viewed in isolation, a component is a ⬆system by itself. |
| 클래스 | **Class** | A representation of a set of ⬆objects of the same kind by describing the structure of the objects, the ways they can be manipulated and how they behave. |
| 클래스 다이어그램 | **Class diagram** | A diagrammatic representation of a ⬆class model. |
| 클래스 모델 | **Class model** | A model consisting of a set of ⬆classes and relationships between them. |
| (요구사항의) 타당성 | Feasibility (of a requirement) | The degree to which a ⬆requirement for a ⬆system can be implemented under existing ⬆constraints. |
| 타임박스 | Timebox | A fixed, non–extendable amount of time for completing a set of ⬆tasks. |
| 탐색적 프로토타입 | **Exploratory prototype** | A throwaway ⬆prototype used to create shared understanding, clarify ⬆requirements or validate requirements. |
| 작업 | Task | A coherent chunk of work to be done. |
| 테마 | Theme | In agile development: A collection of related ⬆user stories. |
| 통합 모델링 언어 | **UML** | Abbreviation for Unified Modeling Language, a standardized language for modeling problems or solutions. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 특성 | **Feature** | A distinguishing characteristic of a ⬆system that provides value for ⬆stakeholders.<br>**Note:**<br>A feature typically comprises several ⬆requirements and is used for communicating with ⬆stakeholders on a higher level of abstraction and for expressing variable or optional characteristics. |
| 특성 다이어그램 | Feature diagram | A diagrammatic representation of a ⬆feature model. |
| 특성 모델 | Feature model | A ⬆model describing the variable features of a ⬆product line, including their relationships and dependencies. |
| 퍼소나 | **Persona** | A fictitious character representing a group of ⬆users with similar needs, values and habits who are expected to use a ⬆system in a similar way. |
| 표준 | **Standard** | A formal, possibly mandatory set of regulations for how to interpret, develop, manufacture, or execute something.<br>**Note:**<br>In RE, there are RE-relevant standards issued by ISO/IEC and IEEE. |
| 품질 | **Quality** | ▪ In general:<br>  ▪ The degree to which a set of inherent characteristics of an item fulfills ⬆requirements.<br>▪ In systems and software engineering:<br>  ▪ The degree to which a ⬆system satisfies stated and implied needs of its ⬆stakeholders.<br>**Note:**<br>Quality in this definition means fitness for intended use, as stated in the ⬆requirements. This is in contrast to the colloquial notion of quality which is typically connoted with goodness or excellence. |
| 품질 요구사항 | **Quality requirement** | A ⬆requirement that pertains to a quality concern that is not covered by ⬆functional requirements. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 제품 라인 | Product line | A jointly managed set of systems (provided as products or services) that share a common core and have a configurable set of ⬆variants for satisfying needs of particular ⬆customers or market segments.<br>**Note:**<br>The points in a product line where there is more than one ⬆variant to select from are called ⬆variation points.<br>**Synonym:** Product family |
| 프로세스 | **Process** | A set of interrelated ⬆activities performed in a given order to process information or materials.<br>**Note:**<br>The notion of process includes *business processes* (e.g., how to commission and send ordered goods to ⬆customers), *information processes* (e.g., how to deliver records from a database that match a given query), and *technical processes* (e.g., cruise control in a car). |
| 프로세스 모델 | **Process model** | A ⬆model describing a ⬆process or a set of related processes. |
| 프로세스 패턴 | Process pattern | An abstract, reusable ⬆model of a ⬆process which can be used to configure and instantiate a concrete process for a given situation and ⬆context. |
| 프로토타이핑 (시제품화) | Prototyping | A ⬆process that involves the creation and evaluation of ⬆prototypes. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| **프로토타입** | **Prototype** | <ul><li>In manufacturing:<ul><li>A piece which is built prior to the start of mass production.</li></ul></li><li>In software and systems engineering:<ul><li>A preliminary, partial realization of certain characteristics of a ⬆system.</li></ul></li><li>In design:<ul><li>A preliminary, partial instance of a design solution.</li></ul></li></ul>**Note:**<ul><li>In RE, prototypes are used as a means for requirements ⬆elicitation (see ⬆specification by example) and ⬆validation.</li><li>Prototypes in RE can be classified<ul><li>with respect to their degree of fidelity into ⬆native prototypes, ⬆mock-ups and ⬆wireframes;</li><li>with respect to their purpose into ⬆exploratory prototypes and ⬆evolutionary prototypes.</li></ul></li></ul> |
| (요구사항의) **필요성** | **Necessity** (of a requirement) | The degree to which an individual ⬆requirement is a necessary part of the ⬆requirements specification of a ⬆system. |
| 항목 | Item | Anything which is perceivable or conceivable.<br>**Synonyms:** entity, object |
| 협상 | Negotiation | → Requirements negotiation |
| 협회 | Association | In UML: A relationship between two ⬆classes in a ⬆UML ⬆class model. |
| **형상** | **Configuration** | A consistent set of logically coherent ⬆items. The items are individually identifiable ⬆work products or parts of work products in at most one ⬆version per item. |
| 활동 | Activity | An action or a set of actions that a person or group performs to accomplish a ⬆task. |
| **활동 다이어그램** | **Activity diagram** | A diagram type in ⬆UML which models the flow of actions in some part of a ⬆system, including ⬆data flows and areas of responsibility where necessary. |
| **활동 모델** | **Activity model** | A ⬆model of the flow of actions in some part of a ⬆system. |

| 용어 (한국어) | Term (English) | Definition |
|---|---|---|
| 효과 | **Effectiveness** | The degree to which an ↑item produces the intended results.<br>**Note:**<br>In RE, effectiveness frequently is the degree to which a ↑system enables its ↑users to achieve their ↑goals. |
| 효율 | **Efficiency** | The degree to which resources are expended in relation to results achieved. |

# 2 Englisch - Korean Dictionary

| Term (English) | 용어 (한국어) |
| --- | --- |
| Acceptance | 인수 |
| **Acceptance criteria** | **인수 조건** |
| Acceptance test | 인수 테스트 |
| Activity | 활동 |
| **Activity diagram** | **활동 다이어그램** |
| **Activity model** | **활동 모델** |
| **Actor** | **액터** |
| **Adequacy** (of a requirement) | (요구사항의) **적절성** |
| **Agile** | **애자일** |
| Ambiguity | 모호성 |
| **Application domain** | **애플리케이션 도메인** |
| Artifact | 산출물 |
| Association | 협회 |
| **Attribute** | **속성** |

| Term (English) | 용어 (한국어) |
| --- | --- |
| Backlog | 백로그 |
| **Baseline** | **기준** |
| **Behavior** | **동작** |
| **Behavior model** | **동작 모델** |
| Branch | 분기 |
| Bug | 버그 |
| Burndown chart | 번다운 차트 |
| **Business requirement** | **비즈니스 요구사항** |
| Cardinality | 카디널리티 |
| **Change control board** | **변경 제어 위원회** |
| **Change management** | **변경 관리** |
| **Change request** | **변경 요청** |
| Changeability | 변경 가능성 |
| **Class** | **클래스** |
| **Class diagram** | **클래스 다이어그램** |

| Term (English) | 용어 (한국어) |
| --- | --- |
| **Class model** | **클래스 모델** |
| Commonality | 공통점 |
| **Completeness** (of requirements) | (요구사항의) **완성도** |
| **Compliance** | **규정 준수** |
| **Component** | **컴포넌트** |
| Composition (in a technical context) | (기술적 맥락에서의) 구성 |
| **Configuration** | **형상** |
| Conflict (about requirements) | (요구사항에 대한) 충돌 |
| **Conformity** | **적합성** |
| **Consistency** (of requirements) | (요구사항의) **일관성** |
| **Constraint** (in RE) | (요구사항에서의) 제약사항 |
| **Context** | **컨텍스트**(정황) |
| **Context boundary** | **컨텍스트**(정황) **경계** |
| **Context diagram** | **컨텍스트 다이어그램** |
| **Context model** | **컨텍스트 모델** |

| Term (English) | 용어 (한국어) |
|---|---|
| Control flow | 제어 흐름 |
| Correctness | 정확성 |
| **Customer** | **고객** |
| **Customer requirements specification** | **고객 요구사항 명세** |
| Data flow | 데이터 흐름 |
| Data flow diagram | 데이터 흐름 다이어그램 |
| Data flow model | 데이터 흐름 모델 |
| Decision table | 결정 테이블 |
| **Defect** | **결함** |
| **Design** | **설계** |
| Document template | 문서 템플릿 |
| **Domain** | **도메인** |
| **Domain model** | **도메인 모델** |
| **Domain requirement** | **도메인 요구사항** |
| **Effectiveness** | **효과** |

| Term (English) | 용어 (한국어) |
|---|---|
| **Efficiency** | **효율** |
| **Elaboration** (of requirements) | (요구사항의) **상세화** |
| **Elicitation** (of requirements) | (요구사항의) **도출** |
| End user | 최종 사용자 |
| Entity | 개체 |
| Entity-relationship diagram | 개체-관계 다이어그램 |
| Entity-relationship model | 개체-관계 모델 |
| Epic | 에픽 |
| **Error** | **에러** |
| **Evolutionary prototype** | **진화적 프로토타입** |
| **Exploratory prototype** | **탐색적 프로토타입** |
| **Fault** | **오류** |
| **Fault tolerance** | **오류 내구성** |
| Feasibility (of a requirement) | (요구사항의) 타당성 |
| **Feature** | **특성** |

| Term (English) | 용어 (한국어) |
|---|---|
| Feature diagram | 특성 다이어그램 |
| Feature model | 특성 모델 |
| Form template | 서식 |
| **Functional requirement** | **기능 요구사항** |
| **Functionality** | **기능성** |
| **Glossary** | **용어집** |
| **Goal** | **목표** |
| **Goal model** | **목표 모델** |
| Homonym | 동음이의어 |
| **Increment** (in software development) | (소프트웨어 개발에서의) **증분** |
| **Inspection** | **인스펙션** |
| Item | 항목 |
| **Iteration** | **반복** |
| **Kind of requirement** | **요구사항의 종류** |
| Language | 언어 |

| Term (English) | 용어 (한국어) |
| --- | --- |
| Maintainability | 유지보수성 |
| Method | 방법 |
| Methodology | 방법론 |
| **Mock-up** (of a digital system) | (디지털 시스템의) **목업** |
| **Model** | **모델** |
| **Modeling language** | **모델 언어** |
| **Modifiability** | **수정 가능성** |
| Multiplicity | 다중도 |
| **Native prototype** | **네이티브 프로토타입** |
| **Natural language** | **자연어** |
| **Necessity** (of a requirement) | (요구사항의) **필요성** |
| Negotiation | 협상 |
| **Non-functional requirement** | **비기능 요구사항** |
| **Object** | **객체** |
| Object diagram | 객체 다이어그램 |

| Term (English) | 용어 (한국어) |
|---|---|
| **Object model** | **객체 모델** |
| **Performance requirement** | **성능 요구사항** |
| **Persona** | **퍼소나** |
| Phrase template | 구문 템플릿 |
| Portability | 이동성 |
| Practice | 실천법 |
| Prioritization | 우선순위 지정 |
| **Priority** | **우선순위** |
| Problem | 문제 |
| **Process** | **프로세스** |
| **Process model** | **프로세스 모델** |
| Process pattern | 프로세스 패턴 |
| **Product** (in the context of software) | (소프트웨어에서의) **제품** |
| **Product backlog** | **제품 백로그** |
| Product line | 제품 라인 |

| Term (English) | 용어 (한국어) |
| --- | --- |
| Product owner | 제품 소유자 |
| **Prototype** | **프로토타입** |
| Prototyping | 프로토타이핑 (시제품화) |
| **Quality** | **품질** |
| **Quality requirement** | **품질 요구사항** |
| **Redundancy** | **중복** |
| Refactoring | 리팩토링 |
| **Release** | **릴리스** |
| **Reliability** | **신뢰성** |
| **Requirement** | **요구사항** |
| Requirements analysis | 요구사항 분석 |
| Requirements baseline | 요구사항 베이스라인 |
| Requirements branching | 요구사항 분기 |
| Requirements configuration | 요구사항 구성 |
| **Requirements conflict** | **요구사항 충돌** |

| Term (English) | 용어 (한국어) |
|---|---|
| Requirements discovery | 요구사항 식별 |
| Requirements document | 요구사항 문서 |
| Requirements elicitation | 요구사항 도출 |
| Requirements Engineer | 요구사항 엔지니어 |
| Requirements Engineering | 요구공학 |
| Requirements management | 요구사항 관리 |
| Requirements model | 요구사항 모델 |
| Requirements negotiation | 요구사항 협상 |
| Requirements source | 요구사항 출처 |
| Requirements specification | 요구사항 명세 |
| Requirements template | 요구사항 템플릿 |
| Review | 리뷰 |
| Risk | 리스크 |
| Role | 역할 |
| Safety | 안전 |

| Term (English) | 용어 (한국어) |
| --- | --- |
| Scenario | 시나리오 |
| **Scope** (of a system development) | (시스템 개발의) **범위** |
| Scrum | 스크럼 |
| **Security** | **보안** |
| **Semantics** | **의미** |
| Semi-formal | 반-정형적 |
| **Sequence diagram** | **시퀀스 다이어그램** |
| **Service** | **서비스** |
| **Software requirements specification** | **소프트웨어 요구사항 명세** |
| Source (of a requirement) | (요구사항의) 출처 |
| **Specification** | **명세** |
| Specification by example | 예시 명세 |
| Specification language | 명세 언어 |
| Spike | 스파이크 |
| **Sprint** | **스프린트** |

| Term (English) | 용어 (한국어) |
|---|---|
| Sprint backlog | 스프린트 백로그 |
| **Stakeholder** | **이해관계자** |
| **Stakeholder requirement** | **이해관계자 요구사항** |
| **Standard** | **표준** |
| **State machine** | **상태 머신** |
| **State machine diagram** | **상태 머신 다이어그램** |
| **Statechart** | **상태 차트** |
| State–transition diagram | 상태 전이 다이어그램 |
| Steering committee | 운영 위원회 |
| Story (in an RE context) | (요구공학에서의) 스토리 |
| Story map | 스토리 맵 |
| Storyboard | 스토리보드 |
| Structured Analysis | 구조화된 분석 |
| Supplier | 공급자 |
| **Synonym** | **동의어** |

| Term (English) | 용어 (한국어) |
|---|---|
| **Syntax** | **구문** |
| **System** | **시스템** |
| **System boundary** | **시스템 경계** |
| System context | 시스템 정황 |
| **System requirement** | **시스템 요구사항** |
| **System requirements specification** | **시스템 요구사항 명세** |
| Task | 작업 |
| Technique | 기법 |
| Theme | 테마 |
| Timebox | 타임박스 |
| **Tool** (in software engineering) | (소프트웨어 공학에서의) **도구** |
| **Traceability** | **추적성** |
| **UML** | **통합 모델링 언어** |
| **Unambiguity** (of requirements) | (요구사항의) **명확성** |
| **Understandability** | **이해 가능성** |

| Term (English) | 용어 (한국어) |
|---|---|
| Usability | 사용성 |
| Use case | 유스케이스 |
| Use case diagram | 유스케이스 다이어그램 |
| Use case model | 유스케이스 모델 |
| User | 사용자 |
| User requirement | 사용자 요구사항 |
| User story | 사용자 스토리 |
| Validation | 유효성 검증 |
| Variability | 변동성 |
| Variant | 변형 |
| Variation point | 변형점 |
| Verifiability (of requirements) | (요구사항의) 검증 가능성 |
| Verification | 베리피케이션 |
| Version | 버전 |
| View | 뷰 |

| Term (English) | 용어 (한국어) |
| --- | --- |
| Viewpoint | 뷰포인트, 시각 |
| **Vision** (for a system or product) | (시스템이나 제품의) **비전** |
| **Walkthrough** | **워크스루** |
| **Wireframe** | **와이어어프레임** |
| **Work product** | **작업 산출물** |

# 3 Sources

I don't cite sources for individual definitions because I deliberately decided not to compile definitions from various existing sources just by copy-paste, but to carefully re-formulate all definitions consistently and according to today's use.

Several definitions are based on my own work [Gl07], [GlWi07], [Gl19]. Most definitions from the agile domain have been taken from or adapted from the IREB RE@Agile Glossary, which was joint work of the RE@Agile working group and me. The revision of the IREB CPRE Foundation Level syllabus [IREB20] also informed several new or changed definitions.

I consulted numerous international standards when writing the definitions [IEEE610], [IEEE730], [IEEE830], [IEEE1012], [IEEE1028], [ISO9000], [ISO12207], [ISO19770], [ISO20246], [ISO24765], [ISO25000], [ISO25010], [ISO26550], [ISO29148], [ISO42010]. However, as the terminology defined or used in these standards is frequently inconsistent or inadequate for a Requirements Engineering glossary, I did not copy any definitions verbatim from these standards.

Other sources that influenced some definitions are [GaWe89], [My06], [Po10], [St73], and [ZoCo05].

For cross-checking, I also consulted the Merriam-Webster online dictionary (https://www.merriam-webster.com) and Wikipedia (https://en.wikipedia.org).

Below I want to give credit for some definitions that I have taken more or less verbatim from a source or that are joint work with others. The copyright for cited definitions lies with the authors of the cited work. The copyright for joint work lies jointly with the author of this glossary and the persons mentioned.

Tabelle 3.1: Credits

| Term | Reference |
| --- | --- |
| Context boundary | Joint work with Klaus Pohl, Chris Rupp, and Thorsten Weyer, based on [Po10], [PoRu11] and [We10] |
| Functional requirement | Joint work with Klaus Pohl, Chris Rupp, and Thorsten Weyer |
| Model | Joint work with Klaus Pohl and Chris Rupp, based on [PoRu11] |
| Quality requirement | Joint work with Klaus Pohl, Chris Rupp, and Thorsten Weyer, based on definitions in my course notes on Requirements Engineering I |
| Requirements Engineering | Definition is a simplification of a definition that was joint work with Klaus Pohl, Chris Rupp, and Thorsten Weyer |
| Requirements specification | Adapted from Pohl and Rupp [PoRu11] |
| System boundary | Joint work with Klaus Pohl, Chris Rupp, and Thorsten Weyer based on [Po10], [PoRu11] |
| System context | Joint work with Klaus Pohl, Chris Rupp, and Thorsten Weyer based on [Po10], [PoRu11], [We10] |

# 4 References

[GaWe89]    Donald C. Gause and Gerald M. Weinberg (1989). Exploring Requirements: Quality before Design. New York: Dorset House.

[Gl07]    Martin Glinz (2007). On Non-Functional Requirements. 15th IEEE International Requirements Engineering Conference (RE'07), Delhi, India. 21–26.

[GlWi07]    Martin Glinz and Roel Wieringa (2007). Stakeholders in Requirements Engineering (Guest Editors' Introduction). IEEE Software 24(2):18–20.

[Gl19]    Martin Glinz (2019). Requirements Engineering I. Course Notes, University of Zurich. https://www.ifi.uzh.ch/en/rerg/courses/hs19/re-i.html#resources. Last visited August 2020.

[IEEE610]    IEEE Standard Glossary of Software Engineering Terminology. IEEE Std 610.12-1990.

[IEEE730]    IEEE Standard for Software Quality Assurance Processes. IEEE Std 730-2014.

[IEEE830]    IEEE Recommended Practice for Software Requirements Specifications. IEEE Std 830-1998.

[IEEE1012]    IEEE Standard for System, Software, and Hardware Verification and Validation. IEEE Std 1012-2016.

[IEEE1028]    IEEE Standard for Software Reviews and Audits. IEEE Std 1028-2008.

[IREB20]    IREB (2020). Certified Professional for Requirements Engineering – Foundation Level – Syllabus, Version 3.0. https://www.ireb.org/en/downloads/#cpre-foundation-level-syllabus-3-0. Last visited September 2020.

[ISO9000]    Quality Management Systems — Fundamentals and Vocabulary. ISO Standard 9000:2015.

[ISO12207]    Systems and Software Engineering — Software Life Cycle Processes. ISO/IEC/IEEE Standard 12207:2017.

[ISO19770]    Information Technology — IT Asset Management — Part 1: IT Asset Management Systems — Requirements. ISO/IEC Standard 19770-1:2017.

[ISO20246]    Software and Systems Engineering — Work Product Reviews. ISO/IEC Standard 20246:2017

[ISO24765]    Systems and Software Engineering — Vocabulary. ISO/IEC/IEEE Standard 24765:2017.

[ISO25000]    Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE. ISO/IEC Standard 25000:2014.

[ISO25010]   Systems and Software Engineering — Systems and Software Quality
             Requirements and Evaluation (SQuaRE) — System and Software Quality
             Models. ISO/IEC Standard 25010:2011.

[ISO26550]   Software and Systems Engineering — Reference Model for Product Line
             Engineering and Management. ISO/IEC Standard 26550:2015.

[ISO29148]   Systems and Software Engineering — Life Cycle Processes — Requirements
             Engineering. ISO/IEC/IEEE Standard 29148:2018.

[ISO42010]   Systems and Software Engineering — Recommended Practice for
             Architectural Description of Software-Intensive Systems. ISO/IEC Standard
             42010:2007.

[My06]       John Mylopoulos (2006). Goal-Oriented Requirements Engineering: Part II.
             Presentation slides of keynote talk at the 14th IEEE International Requirements
             Engineering Conference (RE'06), Minneapolis, USA.

[Po10]       Klaus Pohl (2010). Requirements Engineering: Fundamentals, Principles, and
             Techniques. Berlin-Heidelberg: Springer.

[PoRu11]     Klaus Pohl, Chris Rupp (2011). Requirements Engineering Fundamentals. Santa
             Barbara, Ca.: RockyNook.

[St73]       Herbert Stachowiak (1973). Allgemeine Modelltheorie. (in German) Wien:
             Springer.

[We10]       Thorsten Weyer (2010). Kohärenzprüfung von Verhaltensspezifikationen gegen
             spezifische Eigenschaften des operationellen Kontexts (in German). PhD
             Dissertation, University of Duisburg-Essen.

[ZoCo05]     Didar Zowghi and Chad Coulin (2005). Requirements Elicitation: A Survey of
             Techniques, Approaches, and Tools. In A. Aurum, C. Wohlin (eds.): Engineering
             and Managing Software Requirements. Berlin: Springer. 19–46.